Thursday, 29 January 2026 2:14 PM

Q1. Which statements correctly remove all records from students but keep table structure?

A. DELETE FROM students;
B. DROP TABLE students;
C. TRUNCATE TABLE students;
D. ALTER TABLE students DELETE;

Q2. Which queries correctly find students whose name contains 'an' anywhere?

A. WHERE name LIKE 'an%'
B. WHERE name LIKE '%an%'
C. WHERE name LIKE '%an'
D. WHERE name = '%an%'

Q3. Which statements about COUNT are correct?

A. COUNT(*) ignores NULL rows
B. COUNT(column) ignores NULL values
C. COUNT(*) counts rows
D. COUNT(column) counts only non-NULL values

Q4. Which queries correctly find department-wise student count?

A.

```
SELECT dept_id, COUNT(*) FROM students;
```

B.

```
SELECT dept_id, COUNT(*) FROM students GROUP BY dept_id;
```

C.

```
SELECT COUNT(*), dept_id FROM students GROUP BY dept_id;
```

D.

```
SELECT dept_id FROM students GROUP BY COUNT(*);
```

Q5. Which queries correctly filter groups, not rows?

A. WHERE COUNT(*) > 3
B. HAVING COUNT(*) > 3
C. WHERE AVG(marks) > 60
D. HAVING AVG(marks) > 60

Q6. Which JOINs can return unmatched rows?

A. INNER JOIN
B. LEFT JOIN
C. RIGHT JOIN
D. CROSS JOIN

Q7. Which queries correctly find students not enrolled in any course?

A.

```
SELECT * FROM students
WHERE student_id IN (SELECT student_id FROM enrollments);
```

B.

```
SELECT * FROM students
WHERE student_id NOT IN (SELECT student_id FROM enrollments);
```

C.

```
SELECT s.*
FROM students s
LEFT JOIN enrollments e ON s.student_id=e.student_id
WHERE e.student_id IS NULL;
```

D.

SELECT * FROM students WHERE EXISTS (SELECT * FROM enrollments);

Q8. Which statements about HAVING are correct?

A. Executes before GROUP BY
B. Can use aggregate functions
C. Filters grouped data
D. Cannot be used without GROUP BY

Q9. Which queries correctly find courses with no enrollments?

A.

SELECT * FROM courses
WHERE course_id IN (SELECT course_id FROM enrollments);
B.

SELECT * FROM courses
WHERE course_id NOT IN (SELECT course_id FROM enrollments);
C.

SELECT c.*
FROM courses c
LEFT JOIN enrollments e ON c.course_id=e.course_id
WHERE e.course_id IS NULL;
D.

SELECT * FROM courses WHERE EXISTS (SELECT * FROM enrollments);

Q10. Which queries correctly find students enrolled in more than one course?

A.

SELECT student_id FROM enrollments;
B.

SELECT student_id FROM enrollments
GROUP BY student_id HAVING COUNT(*) > 1;
C.

SELECT student_id FROM enrollments
GROUP BY student_id HAVING COUNT(course_id) > 1;
D.

SELECT student_id FROM enrollments WHERE COUNT(*) > 1;

Q11. Which statements are valid DML commands?

A. INSERT
B. UPDATE
C. DELETE
D. ALTER

Q12. Which queries correctly find students scoring above average marks?

A.

WHERE marks > AVG(marks)
B.

WHERE marks > (SELECT AVG(marks) FROM students)
C.

GROUP BY marks HAVING marks > AVG(marks)
D.

WHERE marks >= ALL (SELECT marks FROM students)

Q13. Which queries correctly find duplicate student names?

A.

SELECT name FROM students;
B.

SELECT name FROM students GROUP BY name HAVING COUNT(*) > 1;
C.

SELECT DISTINCT name FROM students;
D.

SELECT name FROM students GROUP BY name;

Q14. Which subqueries may return multiple rows?

A. Scalar subquery
B. IN subquery
C. EXISTS subquery
D. Correlated subquery

Q15. Which JOIN returns only matching rows?

A. LEFT JOIN
B. RIGHT JOIN
C. INNER JOIN
D. FULL JOIN

Q16. Which queries correctly find departments having more than 3 students?

A.

SELECT dept_id FROM students GROUP BY dept_id;
B.

SELECT dept_id FROM students
GROUP BY dept_id HAVING COUNT(*) > 3;
C.

SELECT dept_id FROM students WHERE COUNT(*) > 3;
D.

SELECT dept_id FROM students HAVING COUNT(*) > 3;

Q17. Which statements about EXISTS are correct?

A. Returns TRUE or FALSE
B. Returns actual data
C. Stops after first match
D. Faster than IN for large datasets

Q18. Which queries correctly update marks by 10 where marks < 50?

A.

UPDATE students SET marks = marks + 10 WHERE marks < 50;
B.

UPDATE students SET marks + 10 WHERE marks < 50;
C.

UPDATE students SET marks = marks + 10;
D.

UPDATE students WHERE marks < 50 SET marks = marks + 10;

Q19. Which queries correctly find second highest marks?

A.

SELECT MAX(marks) FROM students;
B.

SELECT MAX(marks)
FROM students
WHERE marks < (SELECT MAX(marks) FROM students);
C.

```
SELECT DISTINCT marks
FROM students
ORDER BY marks DESC LIMIT 1,1;
```
D.

```
SELECT marks FROM students WHERE marks = 2;
```

Q20. Which statements about DELETE are correct?

A. Can use WHERE clause
B. Resets auto-increment
C. Can be rolled back
D. Removes table structure

Q21. Which queries correctly find students present in both student_copy and student_copy1?

A.

```
SELECT * FROM student_copy;
```
B.

```
SELECT * FROM student_copy
WHERE student_id IN (SELECT student_id FROM student_copy1);
```
C.

```
SELECT * FROM student_copy
WHERE EXISTS (
  SELECT 1 FROM student_copy1
  WHERE student_copy1.student_id = student_copy.student_id
);
```
D.

```
SELECT * FROM student_copy1;
```

Q22. Which queries correctly find departments with no students?

A.

```
SELECT * FROM departments;
```
B.

```
SELECT * FROM departments
WHERE dept_id NOT IN (SELECT dept_id FROM students);
```
C.

```
SELECT d.*
FROM departments d
LEFT JOIN students s ON d.dept_id=s.dept_id
WHERE s.dept_id IS NULL;
```
D.

```
SELECT * FROM departments WHERE EXISTS (SELECT * FROM students);
```

Q23. Which aggregate functions ignore NULL values?

A. COUNT(*)
B. COUNT(column)
C. SUM
D. AVG

Q24. Which queries correctly insert data from old_student into students?

A.

```
INSERT students SELECT * FROM old_student;
```
B.

```
INSERT INTO students SELECT * FROM old_student;
```
C.

```
INSERT INTO students VALUES (SELECT * FROM old_student);
```

D.

```
INSERT INTO students(student_id,name)
SELECT student_id,name FROM old_student;
```

Q25. Which queries correctly find employees earning more than their manager?

A.
Using self join
B.
Using correlated subquery
C.
Using GROUP BY
D.
Using EXISTS

Q26. Which statements about GROUP BY are correct?

A. Reduces rows
B. Used with aggregate functions
C. Required for all SELECT queries
D. Works after WHERE

Q27. Which queries correctly find members who never issued any book?

A.

```
SELECT * FROM members;
```

B.

```
SELECT * FROM members
WHERE member_id NOT IN (SELECT member_id FROM issue_records);
```

C.

```
SELECT m.*
FROM members m
LEFT JOIN issue_records i ON m.member_id=i.member_id
WHERE i.member_id IS NULL;
```

D.

```
SELECT * FROM members WHERE EXISTS (SELECT * FROM issue_records);
```

Q28. Which statements about TRUNCATE are correct?

A. Cannot be rolled back
B. Fires DELETE triggers
C. Faster than DELETE
D. Resets identity

Q29. Which queries correctly find students whose department exists in departments?

A.

```
SELECT * FROM students;
```

B.

```
SELECT * FROM students
WHERE dept_id IN (SELECT dept_id FROM departments);
```

C.

```
SELECT * FROM students
WHERE EXISTS (
  SELECT 1 FROM departments
  WHERE departments.dept_id = students.dept_id
);
```

D.

```
SELECT * FROM students WHERE dept_id = departments.dept_id;
```

Q30. Which queries correctly find books that are currently issued?

A.

```
SELECT * FROM books;
```
B.

```
SELECT * FROM books
WHERE book_id IN (
  SELECT book_id FROM issue_records WHERE return_date IS NULL
);
```
C.

```
SELECT b.*
FROM books b
JOIN issue_records i ON b.book_id=i.book_id
WHERE i.return_date IS NULL;
```
D.

```
SELECT * FROM books WHERE EXISTS (SELECT * FROM issue_records);
```

Q31. Which queries correctly delete students who are not enrolled in any course?
A.

```
DELETE FROM students;
```
B.

```
DELETE FROM students
WHERE student_id NOT IN (SELECT student_id FROM enrollments);
```
C.

```
DELETE s
FROM students s
LEFT JOIN enrollments e ON s.student_id=e.student_id
WHERE e.student_id IS NULL;
```
D.

```
DELETE FROM students WHERE EXISTS (SELECT * FROM enrollments);
```

Q32. Which statements about ANY and ALL are correct?
A. > ANY means greater than at least one value
B. > ALL means greater than every value
C. ANY works only with =
D. ALL can be used with subqueries

Q33. Which queries correctly find students who scored less than all students of department 10?
A.

```
SELECT * FROM students
WHERE marks < ALL (SELECT marks FROM students WHERE dept_id=10);
```
B.

```
SELECT * FROM students
WHERE marks < ANY (SELECT marks FROM students WHERE dept_id=10);
```
C.

```
SELECT * FROM students WHERE dept_id <> 10;
```
D.

```
SELECT * FROM students WHERE marks < (SELECT marks FROM students WHERE dept_id=10);
```

Q34. Which queries correctly find maximum marks per department?
A.

```
SELECT dept_id, marks FROM students GROUP BY dept_id;
```
B.

```
SELECT dept_id, MAX(marks) FROM students GROUP BY dept_id;
```
C.

```
SELECT MAX(marks) FROM students GROUP BY dept_id;
```
D

D.

```
SELECT dept_id FROM students HAVING MAX(marks);
```

Q35. Which statements about INSERT INTO … SELECT are correct?

A. Can copy data from another table
B. Column count must match
C. Can use WHERE clause
D. Used only for DDL

Q36. Which queries correctly find faculty working in departments with students?

A.

```
SELECT * FROM faculty;
```

B.

```
SELECT * FROM faculty
WHERE dept_id IN (SELECT dept_id FROM students);
```

C.

```
SELECT * FROM faculty f
WHERE EXISTS (
  SELECT 1 FROM students s WHERE s.dept_id=f.dept_id
);
```

D.

```
SELECT * FROM faculty WHERE dept_id NOT IN (SELECT dept_id FROM students);
```

Q37. Which queries correctly find departments having more faculty than average faculty per department?

A.

```
SELECT dept_id FROM faculty GROUP BY dept_id;
```

B.

```
SELECT dept_id
FROM faculty
GROUP BY dept_id
HAVING COUNT(*) >
(
  SELECT AVG(cnt)
  FROM (
    SELECT COUNT(*) cnt FROM faculty GROUP BY dept_id
  ) t
);
```

C.

```
SELECT dept_id FROM faculty WHERE COUNT(*) > AVG(*);
```

D.

```
SELECT dept_id FROM faculty HAVING COUNT(*) > 1;
```

Q38. Which queries correctly find books issued more than once?

A.

```
SELECT book_id FROM issue_records;
```

B.

```
SELECT book_id
FROM issue_records
GROUP BY book_id
HAVING COUNT(*) > 1;
```

C.

```
SELECT DISTINCT book_id FROM issue_records;
```

D.

```
SELECT book_id FROM issue_records GROUP BY member_id;
```

Q39. Which statements about correlated subqueries are correct?

A. Executed once
B. Depends on outer query
C. Executed for each row
D. Cannot reference outer query

Q40. Which queries correctly find members who issued at least one book?

A.

```
SELECT * FROM members;
```

B.

```
SELECT * FROM members
WHERE member_id IN (SELECT member_id FROM issue_records);
```

C.

```
SELECT m.*
FROM members m
WHERE EXISTS (
  SELECT 1 FROM issue_records i WHERE i.member_id=m.member_id
);
```

D.

```
SELECT * FROM members WHERE member_id NOT IN (SELECT member_id FROM issue_records);
```

Q41. Which queries correctly find students who belong to departments having more than 5 students?

A.

```
SELECT * FROM students;
```

B.

```
SELECT * FROM students
WHERE dept_id IN (
  SELECT dept_id FROM students GROUP BY dept_id HAVING COUNT(*) > 5
);
```

C.

```
SELECT * FROM students GROUP BY dept_id HAVING COUNT(*) > 5;
```

D.

```
SELECT * FROM students WHERE COUNT(*) > 5;
```

Q42. Which statements about UPDATE are correct?

A. WHERE clause is mandatory
B. Without WHERE, all rows update
C. It is a DML command
D. It can modify multiple columns

Q43. Which queries correctly find students whose names end with ‘a’?

A. WHERE name LIKE 'a%'
B. WHERE name LIKE '%a'
C. WHERE name LIKE '%a%'
D. WHERE name = 'a%'

Q44. Which queries correctly find courses offered by departments having faculty?

A.

```
SELECT * FROM courses;
```

B.

```
SELECT * FROM courses
WHERE dept_id IN (SELECT dept_id FROM faculty);
```

C.

```
SELECT * FROM courses c
WHERE EXISTS (
  SELECT 1 FROM faculty f WHERE f.dept_id=c.dept_id
  SELECT 1 FROM faculty f WHERE f.dept_id=c.dept_id
);
```
D.

```
SELECT * FROM courses WHERE dept_id NOT IN (SELECT dept_id FROM faculty);
```

Q45. Which statements about DROP are correct?

A. Removes table structure
B. Can be rolled back
C. Deletes data permanently
D. Removes dependent objects

Q46. Which queries correctly find second highest marks?

A.

```
SELECT MAX(marks) FROM students;
```
B.

```
SELECT MAX(marks)
FROM students
WHERE marks < (SELECT MAX(marks) FROM students);
```
C.

```
SELECT DISTINCT marks
FROM students
ORDER BY marks DESC LIMIT 1,1;
```
D.

```
SELECT marks FROM students WHERE marks = 2;
```

Q47. Which queries correctly find departments that have courses but no enrollments?

A.

```
SELECT * FROM departments;
```
B.

```
SELECT dept_id FROM courses
WHERE course_id NOT IN (SELECT course_id FROM enrollments);
```
C.

```
SELECT DISTINCT dept_id
FROM courses
WHERE course_id NOT IN (SELECT course_id FROM enrollments);
```
D.

```
SELECT dept_id FROM departments WHERE EXISTS (SELECT * FROM enrollments);
```

Q48. Which statements about LIKE are correct?

A. % matches multiple characters
B. _ matches exactly one character
C. LIKE is case-sensitive in all DBs
D. LIKE works only with strings

Q49. Which queries correctly find students common in students and old_student?

A.

```
SELECT * FROM students;
```
B.

```
SELECT * FROM students
WHERE student_id IN (SELECT student_id FROM old_student);
```
C.

```
SELECT s.*
FROM students s
WHERE EXISTS (
  SELECT 1 FROM old_student o WHERE o.student_id=s.student_id
```

);
D.

SELECT * FROM students WHERE student_id NOT IN (SELECT student_id FROM old_student);

Q50. Which queries correctly find departments where all students scored above 60?
A.

SELECT dept_id FROM students GROUP BY dept_id;
B.

```
SELECT dept_id
FROM students
GROUP BY dept_id
HAVING MIN(marks) > 60;
```
C.

SELECT dept_id FROM students WHERE marks > 60;
D.

SELECT dept_id FROM students HAVING marks > 60;

Q51. Which queries correctly find students who are enrolled in all courses?
A.

SELECT student_id FROM enrollments GROUP BY student_id;
B.

```
SELECT student_id
FROM enrollments
GROUP BY student_id
HAVING COUNT(DISTINCT course_id) = (SELECT COUNT(*) FROM courses);
```
C.

SELECT student_id FROM students;
D.

SELECT student_id FROM enrollments WHERE course_id = ALL (SELECT course_id FROM courses);

Q52. Which statements about NOT IN with subqueries are correct?
A. Fails if subquery returns NULL
B. Always faster than NOT EXISTS
C. Works only with single-row subquery
D. Needs same datatype comparison

Q53. Which queries correctly find departments with at least one course?
A.

SELECT * FROM departments;
B.

```
SELECT * FROM departments
WHERE dept_id IN (SELECT dept_id FROM courses);
```
C.

```
SELECT d.*
FROM departments d
WHERE EXISTS (
  SELECT 1 FROM courses c WHERE c.dept_id = d.dept_id
);
```
D.

SELECT * FROM departments WHERE dept_id NOT IN (SELECT dept_id FROM courses);

Q54. Which queries correctly find students whose marks are above department average?
A.

```
SELECT * FROM students WHERE marks > AVG(marks);
```
B.

```
SELECT s.*
FROM students s
WHERE marks > (
  SELECT AVG(marks)
  FROM students
  WHERE dept_id = s.dept_id
);
```
C.

```
SELECT * FROM students GROUP BY dept_id HAVING marks > AVG(marks);
```
D.

```
SELECT * FROM students WHERE marks >= ALL (SELECT marks FROM students);
```

Q55. Which statements about CREATE TABLE AS SELECT are correct?
A. Copies table structure only
B. Copies data from SELECT query
C. Can apply WHERE clause
D. Copies constraints automatically

Q56. Which queries correctly find courses that have maximum enrollments?
A.

```
SELECT course_id FROM enrollments;
```
B.

```
SELECT course_id
FROM enrollments
GROUP BY course_id
HAVING COUNT(*) = (
  SELECT MAX(cnt)
  FROM (
    SELECT COUNT(*) cnt
    FROM enrollments
    GROUP BY course_id
  ) t
);
```
C.

```
SELECT course_id FROM courses;
```
D.

```
SELECT course_id FROM enrollments WHERE COUNT(*) = MAX(COUNT(*));
```

Q57. Which queries correctly find members who issued more than one book?
A.

```
SELECT member_id FROM issue_records;
```
B.

```
SELECT member_id
FROM issue_records
GROUP BY member_id
HAVING COUNT(book_id) > 1;
```
C.

```
SELECT DISTINCT member_id FROM issue_records;
```
D.

```
SELECT member_id FROM issue_records WHERE COUNT(*) > 1;
```

Q58. Which statements about DELETE with subquery are correct?
A. DELETE cannot use subquery
B. DELETE can use WHERE with IN
C. DELETE can remove selective rows
D. DELETE always removes all rows

Q59. Which queries correctly find students who never appeared in old_student?
A.

```
SELECT * FROM students;
```
B.

```
SELECT * FROM students
WHERE student_id NOT IN (SELECT student_id FROM old_student);
```
C.

```
SELECT s.*
FROM students s
WHERE NOT EXISTS (
  SELECT 1 FROM old_student o WHERE o.student_id=s.student_id
);
```
D.

```
SELECT * FROM students WHERE EXISTS (SELECT * FROM old_student);
```

Q60. Which queries correctly find departments where number of students is less than average?
A.

```
SELECT dept_id FROM students GROUP BY dept_id;
```
B.

```
SELECT dept_id
FROM students
GROUP BY dept_id
HAVING COUNT(*) < (
  SELECT AVG(cnt)
  FROM (
    SELECT COUNT(*) cnt FROM students GROUP BY dept_id
  ) t
);
```
C.

```
SELECT dept_id FROM students WHERE COUNT(*) < AVG(*);
```
D.

```
SELECT dept_id FROM students HAVING COUNT(*) < AVG(marks);
```

Q61. Which queries correctly find students sharing same department as student_id = 5?
A.

```
SELECT * FROM students WHERE student_id = 5;
```
B.

```
SELECT * FROM students
WHERE dept_id = (
  SELECT dept_id FROM students WHERE student_id = 5
);
```
C.

```
SELECT * FROM students GROUP BY dept_id;
```
D.

```
SELECT * FROM students WHERE dept_id IN (5);
```

Q62. Which statements about UNION are correct?
A. UNION keeps duplicates
B. UNION removes duplicates
C. Column count must match
D. Datatypes must be compatible

Q63. Which queries correctly find books that were never issued?

Q63. Which queries correctly find books that were never issued?
A.

```
SELECT * FROM books;
```
B.

```
SELECT * FROM books
WHERE book_id NOT IN (SELECT book_id FROM issue_records);
```
C.

```
SELECT b.*
FROM books b
LEFT JOIN issue_records i ON b.book_id=i.book_id
WHERE i.book_id IS NULL;
```
D.

```
SELECT * FROM books WHERE EXISTS (SELECT * FROM issue_records);
```

Q64. Which queries correctly find students whose marks are equal to department maximum?
A.

```
SELECT * FROM students WHERE marks = MAX(marks);
```
B.

```
SELECT s.*
FROM students s
WHERE marks = (
  SELECT MAX(marks)
  FROM students
  WHERE dept_id = s.dept_id
);
```
C.

```
SELECT * FROM students GROUP BY dept_id HAVING MAX(marks);
```
D.

```
SELECT * FROM students WHERE marks >= ALL (SELECT marks FROM students);
```

Q65. Which statements about INDEX are correct?
A. Improves read performance
B. Slows down INSERT/UPDATE
C. Automatically created on all columns
D. Consumes storage space

Q66. Which queries correctly find employees earning less than at least one employee of dept 20?
A.

```
SELECT * FROM employees WHERE salary < ALL (SELECT salary FROM employees WHERE dept_id=20);
```
B.

```
SELECT * FROM employees WHERE salary < ANY (SELECT salary FROM employees WHERE dept_id=20);
```
C.

```
SELECT * FROM employees WHERE dept_id <> 20;
```
D.

```
SELECT * FROM employees WHERE salary < (SELECT salary FROM employees WHERE dept_id=20);
```

Q67. Which queries correctly find departments having exactly one student?
A.

```
SELECT dept_id FROM students GROUP BY dept_id;
```
B.

```
SELECT dept_id
FROM students
```

```
GROUP BY dept_id
HAVING COUNT(*) = 1;
```
C.

```
SELECT dept_id FROM students WHERE COUNT(*) = 1;
```
D.

```
SELECT dept_id FROM students HAVING COUNT(*) = 1;
```

Q68. Which statements about ORDER BY are correct?

A. Executes before WHERE
B. Can use column alias
C. Default order is ascending
D. Can sort by column position

Q69. Which queries correctly find students whose marks are greater than ALL students of cities having avg marks < 60?

A.

```
SELECT * FROM students WHERE marks > 60;
```
B.

```
SELECT * FROM students
WHERE marks > ALL (
  SELECT marks FROM students
  WHERE city IN (
    SELECT city FROM students GROUP BY city HAVING AVG(marks) < 60
  )
);
```
C.

```
SELECT * FROM students GROUP BY city HAVING AVG(marks) < 60;
```
D.

```
SELECT * FROM students WHERE city NOT IN ('<60');
```

Q70. Which queries correctly find employees who are managers?

A.

```
SELECT * FROM employees;
```
B.

```
SELECT DISTINCT manager_id FROM employees WHERE manager_id IS NOT NULL;
```
C.

```
SELECT emp_id FROM employees
WHERE emp_id IN (SELECT manager_id FROM employees);
```
D.

```
SELECT emp_id FROM employees GROUP BY manager_id;
```